SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

Fakulta chemickej a potravinárskej technológie

Evidenčné číslo: FCHPT-5415-68780

Flash prezentácie pre predmet Informatizácia a informačné systémy

Bakalárska práca

2013 Michal Kleščík

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

Fakulta chemickej a potravinárskej technológie

Flash prezentácie pre predmet Informatizácia a informačné systémy

Bakalárska práca

Evidenčné číslo: FCHPT-5415-68780

Študijný program: automatizácia, informatizácia a manažment v chémii a potravinárstve

Číslo študijného odboru[1]: 2621

Názov študijného odboru: 5.2.14 automatizácia, 5.2.52 priemyselné inžinierstvo

Školiace pracovisko: Ústav informatizácie, automatizácie a matematiky

Vedúci záverečnej práce: Ing. Ľuboš Čirka, PhD.

Konzultant: Ing. Martin Kalúz

Bratislava 2013 Michal Kleščík

[1] Prvé štvorčíslie kódu podľa vyhlášky Štatistického úradu Slovenskej republiky č. 114/2011 Z.z., ktorou sa vydáva Štatistická klasifikácia odborov vzdelania

Slovenská technická univerzita v Bratislave
Ústav informatizácie, automatizácie a matematiky

Fakulta chemickej a potravinárskej technológie
Akademický rok: 2012/2013
Evidenčné číslo: FCHPT-5415-68780

STU
FCHPT

ZADANIE BAKALÁRSKEJ PRÁCE

Študent:	**Michal Kleščík**
ID študenta:	68780
Študijný program:	automatizácia, informatizácia a manažment v chémii a potravinárstve
Kombinácia študijných odborov:	5.2.14 automatizácia, 5.2.52 priemyselné inžinierstvo
Vedúci práce:	Ing. Ľuboš Čirka, PhD.
Konzultant:	Ing. Martin Kalúz

Názov práce: **Flash prezentácie pre predmet Informatizácia a informačné systémy**

Špecifikácia zadania:

Cieľom práce je vytvoriť prezentácie v Adobe Flash, ktoré budú použité ako doplnkové výukové materiály v predmete Informatizácia a informačné systémy.

Úlohy:
- Naštudovať problematiku tvorby webových stránok (HTML, CSS, JavaScript).
- Naštudovať program Adobe Flash a skriptovací jazyk ActionScript.
- Programovo realizovať prezentácie vo Flash a ActionScript.
- Implementovať vytvorené prezentácie do LMS Moodle a overiť ich funkčnosť.

Rozsah práce: 30

Zoznam odbornej literatúry:

1. KERMAN, P. *ActionScript ve Flashi : Podrobná příručka.* Praha: Computer Press, 2002. 507 s. ISBN 80-7226-615-2.
2. KELEMEN, S. – STUCHLÍKOVÁ, Ľ. Adobe Flash ako efektívny nástroj popularizácie vedy a techniky. *Posterus [elektronický zdroj] : Internetový časopis Vol. 4, Iss. 8.* ISSN 1338-0087.
3. ADOBE CREATIVE TEAM. *Adobe Flash CS3 : Oficiální výukový kurz.* Brno: Computer Press, 2008. 341 s. ISBN 978-80-251-2109-2.
4. DRUSKA, P. *CSS a XHTML tvorba dokonalých webových stránek krok za krokem.* Praha: Grada Publishing, 2006. 200 s. ISBN 80-247-1382-9.
5. STANÍČEK, P. *CSS hotová řešení.* Brno: Computer Press, 2003. 263 s. ISBN 80-251-1031-1.
6. GRÖPL, T. *HTML, CSS a JavaSript : referenční příručka.* Praha: BEN – technická literatura, 2002. 157 s. ISBN 80-7300-099-7.
7. ŠKULTÉTY, R. *JavaScript : Programujeme internetové aplikace.* Praha: Computer Press, 2004. 224 s. ISBN 80-251-0144-4.

Riešenie zadania práce od: 13. 02. 2013
Dátum odovzdania práce: 19. 05. 2013

Michal Kleščík
študent

prof. Ing. Miroslav Fikar, DrSc.
vedúci pracoviska

prof. Ing. Miroslav Fikar, DrSc.
garant študijného programu

Poďakovanie

Týmto by som chcel poďakovať svojmu konzultantovi Ing. Martinovi Kalúzovi a vedúcemu práce Ing. Ľubošovi Čirkovi, PhD. za pomoc a cenné rady pri písaní bakalárskej práce.

Abstrakt:

Bakalárska práca je zameraná na tvorbu prezentácií v Adobe Flash, ktoré budú použité na lepšie pochopenie učiva v predmete Informatizácia a informačné systémy. Práca je rozdelená na dve časti. Teoretická časť sa zaoberá súčasnou situáciou vzdelávania formou interaktívnych prezentácií a ich návrhom pomocou rôznych dostupných prostriedkov. Praktická časť sa venuje konkrétnemu návrhu a realizácii interaktívnych pomôcok pri štúdiu zvolenej témy.

Kľúčové slová: Adobe Flash, interaktívne prezentácie, DiV

Abstract:

This bachelor's degree thesis is focused on creating of presentations in Adobe Flash. These presentations should later serve for learning purposes of students that have signed up for Informatics & Information Systems class. Theoretical part of the thesis discusses current trends of dynamic learning system through interactive presentations and their construction with various available tools. Practical part of the thesis is focused solely on design and realization of given topic.

Key words: Adobe Flash, interacitive presentations, DiV

Obsah

Zoznam obrázkov

Zoznam príloh

Príloha A: CD médium: práca v elektronickej podobe, prílohy v elektronickej podobe.

Úvod

Súčasná doba ponúka množstvo príležitostí pre vzdelávanie či už klasickou formou alebo formou využitia internetu a informačných technológií. Internet ako zdroj dát a informácií ponúka ľuďom možnosť vzdelávať sa v rôznych odvetviach či už technologických alebo sociálnych. Elektronické vzdelávanie je v posledných rokoch čoraz častejšie preferované, ako zo strany vyučujúcich, tak aj študentov. Školy mnohokrát ponúkajú svojim študentom on-line kurzy, ktoré by mali zabezpečiť neobmedzený prístup k študijným materiálom. Rôzne on-line kurzy bývajú v rámci jedného vzdelávacieho prostredia často centralizované prostredníctvom softvéru typu LMS (angl. Learning Management System), ktorý zabezpečuje správu užívateľov, kurzov, a ich obsahu.

Jedným z najviac používaných systémov LMS je v dnešnej dobe Moodle. Moodle, modulárne objektovo orientované dynamické výukové prostredie, je softvérový systém slúžiaci na vytváranie rôznych kurzov založených na web stránkach dostupných cez Internet. Jedná sa o projekt, ktorý neustále napreduje a je určený na podporu spoločenského aj individuálneho rámca vzdelávania.

Pre tvorbu podporných vyučovacích materiálov akými sú prezentácie a animácie ponúkajú dnešné technológie široký výber dostupných programov, z ktorých si vývojár môže individuálne vybrať. Najčastejšie používané technológie pre tvorbu vizualizácií a animácií v dnešnej dobe sú Webové technológie štandardizované W3C konzorciom (HTML, XML, CSS, SVG, a iné) a technológie umožňujúce vytváranie vnáraného Webového obsahu, akými sú Adobe Flash, Java, MS Silverlight, a iné. Medzi najznámejšie skriptovacie jazyky patria PHP, ASP, Perl a Python na strane servera a JavaScript na strane klienta. Adobe Flash je softvérová platforma slúžiaca na tvorbu interaktívnych aplikácií, videí, ale aj na vytváranie prezentácií pre vyučovanie. Veľkou výhodou takýchto programov a aplikácií je ich nezávislosť na type operačného systému počítača, keďže sú vykonávané prostredníctvom prehliadača Flash Player, ktorý je dostupný pre všetky bežné operačné systémy. Ak je Flash prvok na webovej stránke, tak sa po jeho načítaní stiahne do klientovho počítača a tam sa spúšťa. Aj jeho kompatibilita s webovými prehliadačmi poskytuje široký výber pre klienta. Flash je vhodný aj pre ľudí, ktorí nemajú veľké skúsenosti s programovaním, pretože veľa operácií sa dá vykonávať aj ručne napr. skladanie animácií. Ale asi tou najväčšou výhodou je jeho formát SWF, ktorý je objemovo malý, a tým pádom aj rýchlo načitateľný a ľahký pre kompresiu. Cieľom práce je vytvorenie výukových interaktívnych materiálov pre predmet informácie a informačné systémy pre lepšie pochopenie prednášaného učiva.

2. Teoretická časť

2.1 Dištančné vzdelávanie

Dištančné vzdelávanie (DiV – distance education) je multimediálna forma riadeného štúdia, v ktorom sú vyučujúci a konzultanti v priebehu vzdelávania trvale alebo prevažne oddelení od vzdelávaných [1]. Pre nich je k dispozícii multimediálne využitie všetkých dostupných dištančných komunikačných prostriedkov, pomocou ktorých by vyučujúci mohli učivo efektívne prezentovať napr. PC programy na CD alebo DVD, e-mail, internet alebo aj priama komunikácia cez Skype.

Hlavným cieľom by malo byť, čo najkvalitnejšie poskytnutie multimediálnych informácií pre študentov aj keď základným študijným materiálom ostávajú stále tlačené texty, ktoré sú však odlišné od textov určených pre prezenčnú formu štúdia. Využívané texty obsahujú množstvo otázok, cvičení, krátkych testov, zhrnutí atď. Cieľom je, aby študent mohol využiť tieto texty pre samostatné štúdium, no je ich možné využiť aj pre ďalšie formy štúdia ako napr. prezenčná, diaľková a pod.

Dištančné štúdium umožňuje nadobudnutie nových vedomostí či už pre študentov, ktorí sa nemohli zúčastniť pri preberaní danej látky alebo chcú hlbšie pochopiť problematiku daného predmetu.

2.1.1 Vzťah DiV a prezenčnej formy štúdia

Tradičná denná forma štúdia formou prednášky/cvičení vychádza z priameho kontaktu študenta s učiteľom a je charakteristická tým, že [1]:

a) riadi sa platnými osnovami bez ohľadu na znalosti študentov
b) je bezprostredne sprostredkovaná učiteľom

Dištančná forma je charakterizovaná tým, že [1]:

a) študijné programy sú zamerané na cieľovú skupinu študentov a sú tvorené odborníkmi na danú problematiku
b) sú využívané multimediálne prostriedky (pozn. predpokladá sa, že väčšina študentov má ku nim prístup)

c) je sprostredkovaná počítačmi, ktoré umožňujú študentom pracovať so študijnými materiálmi a zároveň pri problémoch aj komunikovať s vyučujúcim.

2.1.2 Výhody a nevýhody dištančného vzdelávania

V poslednom rade, by som rád ukázal pár výhod aj nevýhod vyučovania touto formou.

Za výhody môžeme považovať:

- malé nároky na učebne a s tým spojené prevádzkové náklady
- flexibilita modelov štúdia a možnosť interaktívnych postupov pri učení
- väčšia možnosť testovania znalostí
- ľahší a rýchlejší prístup k informáciám, vyššia miera interaktivity (moduly obsahujú mnoho multimediálnych prvkov, ktoré zvyšujú dynamickosť). Účinná na tvorbu interaktivity je simulácia aj keď môže byť niekedy dosť zložitá

Medzi nevýhody patria:

- nutnosť zaškolenia pedagogických pracovníkov pre novú technológiu
- potreba spracovať dištančné vzdelávacie opory
- zvýšené náklady na technické vybavenie inštitúcie a prípravu, ale aj zakúpenie softwarových programov
- nutnosť založenia a financovania organizačného pracoviska pre dištančné štúdium
- vysoká miera izolovanosti študujúceho, ktorá môže byť u niektorých študujúcich veľkým problémom

2.2 Technológie využívané pre tvorbu webových stránok

V dnešnej dobe je už takmer nepredstaviteľné aby nejaká firma, subjekt, ktorý chce byť úspešný v boji na trhu plnom konkurentov, nemala vlastnú webovú stránku na internete. Existuje mnoho firiem, ale aj jednotlivcov, ktorý ponúkajú návrh a následné realizovanie stránok pre podniky, firmy, vzdelávacie inštitúcie a pod.. V tejto časti by som rád spomenul niekoľko najpoužívanejších jazykov, ktoré slúžia pre tvorbu webových stránok.

2.2.1 Tvorba pomocou HTML

[2], [3]

HTML (Hypertext Markup Language) je jedným z jazykov pre vytváranie stránok pre World Wide Web, ktorý umožňuje publikáciu dokumentov na Internete. Vznikol v roku 1990 ako potreba vytvorenia jednoduchšej verzie pre tvorbu dokumentov.. Jazyk html prešiel niekoľkými verziami (0.9, 2.0, 3.2, 4.01, 5.0).

Verzia 5

Momentálne je v štádiu návrhu organizáciou W3C. Bude charakteristická novými značkami, podporou offline aplikácií, úložiskom formou asociatívneho poľa. Aj napriek tomu, že HTML 5 ešte nie je plne podporovaný webovými prehliadačmi a je stále vo vývoji, tak už sa v značnej miere používa a dá sa charakterizovať ako technológia pre tvorbu internetových aplikácií konkurujúca Adobe Flash, Jave a MS Silverlight. Hlavným cieľom HTML5 je lepšia priehľadnosť zdrojového kódu. Stránka má určitú vlastnú štruktúru tvorenú časťami ako hlavička, stĺpce a päta (Obr. 1).

```
<!DOCTYPE html>
<html>
  <head>
    <title> Názov dokumentu </title>
          <!-- Komentár: hlavička môže obsahovať doplňujúce informatívne značky -->
  </head>
  <body>
        <!-- Komentár: hlavička môže obsahovať doplňujúce informatívne značky -->
  </ body>
</html>
```

Obr. 1: Príklad HTML

Doctype vyjadruje deklaráciu typu dokumentu, to znamená aký typ dokumentu sa bude spracúvať. Ďalej môžeme vidieť hlavičku dokumentu *head*, kde sa väčšinou definujú podmienky platné pre celý dokument. *Title* je pomenovanie názvu stránky a potom už nasleduje samotné telo dokumentu *body*. Na záver by sa mala dať stránka validovať, aby boli dodržané určité štandardné podmienky pri tvorbe stránok.

2.2.2 CSS

CSS (*Cascading Style Sheet)* je jazyk pre vizuálne formátovanie vzhľadu stránky napísanej v HTML, XHTML alebo XML. Je navrhnutý štandardizačnou organizáciou W3C. Hlavným cieľom je umožniť vývojárom oddeliť vzhľad dokumentu od jeho štruktúry a obsahu. Čo sa týka syntaxe, pozostáva z niekoľkých pravidiel. Každé pravidlo obsahuje selektor a blok deklarácií. Zoznamy deklarácií sú oddelené bodkočiarkou a každá deklarácia pozostáva z názvu vlastnosti a hodnoty vlastnosti.

```
/* CSS  */
body { background: url(pozadie.jpg) }
div { width: 800px; height: 200px; }
#id { background-color: white;}
.class { display: block; color: green; text-decoration: underline;}
```

Obr. 2: Príklad CSS

Na Obr. 2 je uvedený príklad zápisu CSS pomocou selektorov. Selektor *body* odkazuje na značku rovnakého názvu, teda na element definujúci telo dokumentu. Obsah v zložených zátvorkách definuje vlastnosti CSS. Každá definícia vlastnosti pozostáva z názvu a hodnoty vlastnosti oddelených dvojbodkou. V prípade ak jeden selektor má viac vlastností, oddeľujeme ich bodkočiarkou. V rámci dokumentu je možné sa pomocou selektora v tvare *#id* odkázať na konkrétny element, ktorý má takýto identifikátor zadefinovaný. V prípade, že sa potrebujeme selektorom odkázať na viacero elementov v dokumente môžeme im prideliť názov jednej triedy (napr. *class*) a ako selektor použiť tvar *class*.

2.2.3 JavaScript

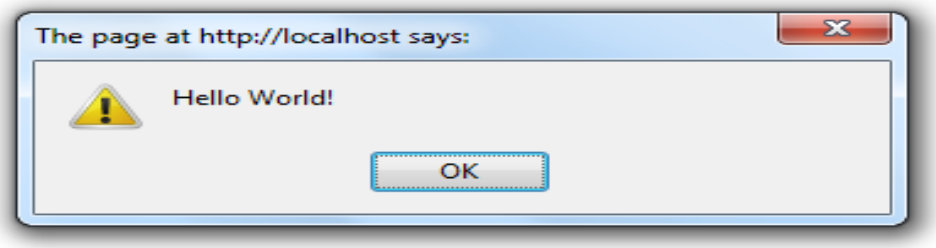
[4]

JavaScript je skriptovací jazyk interpretovaný na strane webového prehliadača, slúžiaci k tvorbe interaktívnych webových stránok. Ide o multiplatformu, objektovo orientovaný jazyk vytvorený spoločnosťou Netscape v roku 1995. Jeho veľkou výhodou je, že sa vkladá priamo do HTML kódu podobne ako CSS štýl, buď ako externý súbor alebo ako interný skript (Obr. 3). Ďalšou výhodou je, že ide o klientský skriptovací jazyk to znamená, že pre jeho

fungovanie netreba server, ale všetko funguje na počítači užívateľa. Má výbornú podporu v prehliadačoch, čoho výsledkom je aj jeho časté využívanie.

```
/* JavaScript */
<script type="text/javascript">
        document.write(“Hello, World!”);
</script>
You can also pop alert boxes just as easily with:
<script type="text/javascript">
        document.alert(“Hello, World!”);
</script>
```

Obr. 3: Príklad JavaScriptu



Obr. 4: Vytvorenie jednoduchého okna

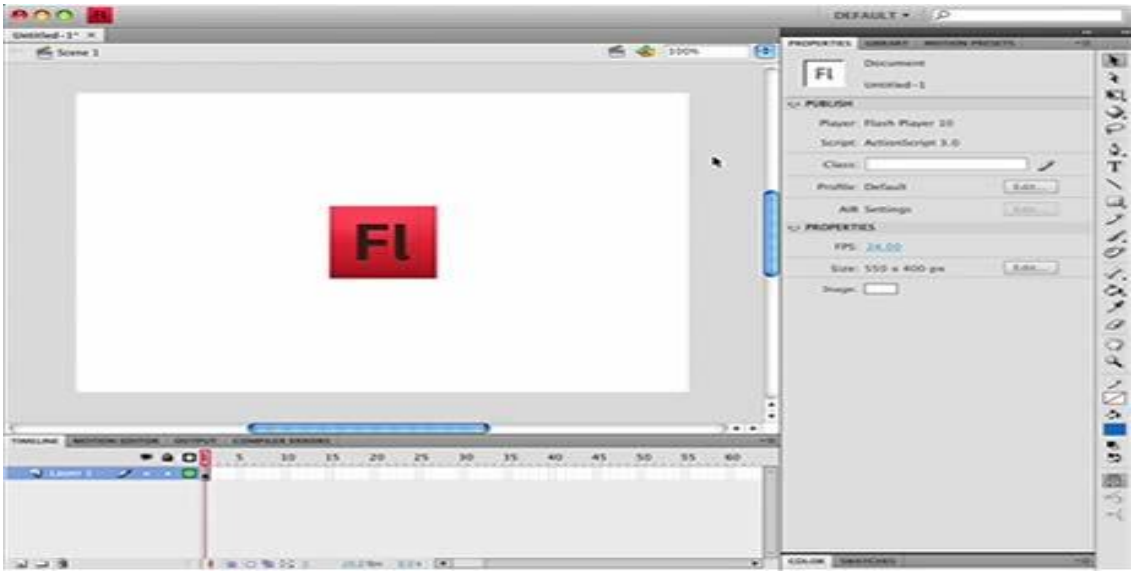
JavaScript slúži na ovládanie rôznych interaktívnych prvkov (Obr. 4), tlačidiel, posúvníkov alebo aj na tvorbu animácií a pod.. Je vlastne programom slúžiacim pri zmene vlastností a obsahu elementov dynamického webu.

2.3 Flash a jeho jazykové prostriedky

[5], [6], [7]

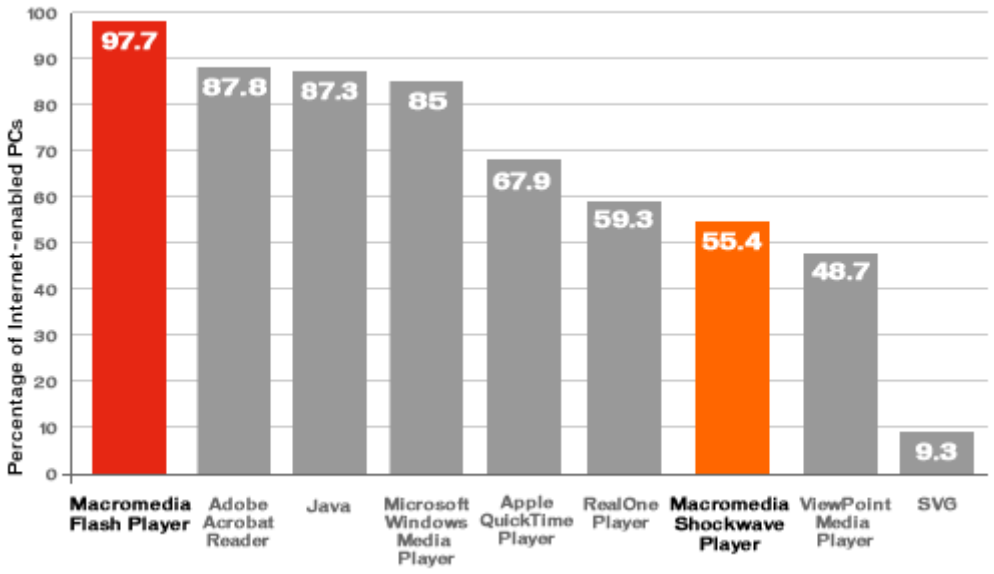
Adobe Flash je multimediálna platforma, ktorá je profesionálnym štandardom pre vytváranie dynamických aplikácií. Používa sa predovšetkým na tvorbu multimediálnych interaktívnych webových stránok (Obr. 5).

Medzi jeho ďalšie prednosti patrí napr. možnosť narábania s audio a video súbormi a tiež možnosť využitia na úpravu fotografií alebo aj ako nástroj pre tvorbu webových hier. K rozšíreniu platformy pomohla vo veľkej miere aj výsledná malá veľkosť súborov, ktorá je odrazom využívania vektorového formátu pri ktorom sú využité matematické výpočty..



Obr. 5: Prostredie Adobe Flash CS4 Professional

Flashové aplikácie môžeme používať na rôznych zariadeniach a operačných systémoch, ktoré využívajú Adobe Flash Player. Je doplnkom mnohých prehliadačov a aj iných multimediálnych zariadení napr. mobilov. Má vysoký percentuálny podiel využitia na webe ako možno vidieť na Obr. 6.[6]



Obr. 6: Podiel softwarových technológií na internete z roku 2010

2.3.1 História

Predchodca Flashových aplikácií bol SmartSketch, ktorý fungoval na Java štruktúre. Keďže vývoj internetu napredoval a využívanie Java ako pôvodného jazyka už nebolo vyhovujúce podmienkam, došlo k totálnej zmene podoby, k čomu prispelo aj využívanie zásuvných modulov tzv. PLUG-IN v prehliadačoch. Bol uvedený pod novým názvom FutureSplash Animator vhodný pre rôzne platformy. Neskôr ho kúpila spoločnosť Macromedia a tak bola vytvorená prvá verzia v roku 1996 pod názvom Macromedia Flash 1.0. Teraz je však vlastníkom firma Adobe Systems.

2.3.2 Výhody a nevýhody Flash technológie

[7]

Medzi základné výhody Flash môžeme radiť:

- interaktivita Flash môže reagovať na veľké množstvo udalostí vyvolaných užívateľom, alebo programom samotným. Medzi takéto patria napr. pohyby myšou, stlačenie kláves a pod.
- aj keď je SWF starší formát, jeho prehliadač je rozšírenejší a lepšie optimalizovaný na rýchlosť
- podporuje streaming videa a zvukov

- kompatibilita aj s externými súbormi
- vďaka tomu, že Flash plug-in sa vyvíja v tesnej väzbe na grafickú aplikáciu Adobe Flash, môže sa grafik vždy spoľahnúť na dokonalú funkčnosť na všetkých podporovaných platformách, pokiaľ práve užívateľ nie je s inštaláciou oneskorený o jednu vývojovú verziu plug-inu
- v neposlednom rade je výhodou aj jeho veľká rozšírenosť a ľahká dostupnosť

Nevýhody Flash technológie

- hlavnou nevýhodou je veľká náročnosť na hardvér
- formát je uzatvorený a autori sú odkázaní na aplikácie, ktoré export SWF podporuje
- vývojové prostredie Adobe Flash je komerčného charakteru (treba ho zakúpiť) a v súčasnosti neexituje voľne dostupná alternatíva s takou komplexnou funkcionalitou

2.3.3 Vektorová grafika

[8]

Je založená na princípe, že obrázok je zložený z vektorov, kriviek spojujúcich tzv. hlavné body. Tieto krivky môžu mať farebnú výplň formou jednoliatej plochy alebo farebného prechodu (gradientov). Základom vektorovej grafiky je matematika. V 70. rokoch francúzsky matematik a konštruktér Pierre Béziere vyvinul matematickú metódu, na ktorej bol schopný popísať ľubovoľný úsek krivky len za pomoci štyroch bodov [9]. Stačí len poznať dva krajné tzv. hlavné body, ktoré definujú danú úsečku a dva tzv. kontrolné body určujúce vlastný tvar krivky. Spojnice medzi kontrolným a hlavným bodom je dotyčnicou k výslednej krivke. Týmto spôsobom môžeme popísať aj tú najzložitejšiu krivku akú sme schopní nakresliť. Krivka nám vytvorí cestu, ktorá môže byť otvorená alebo zatvorená s výplňou, či bez nej.

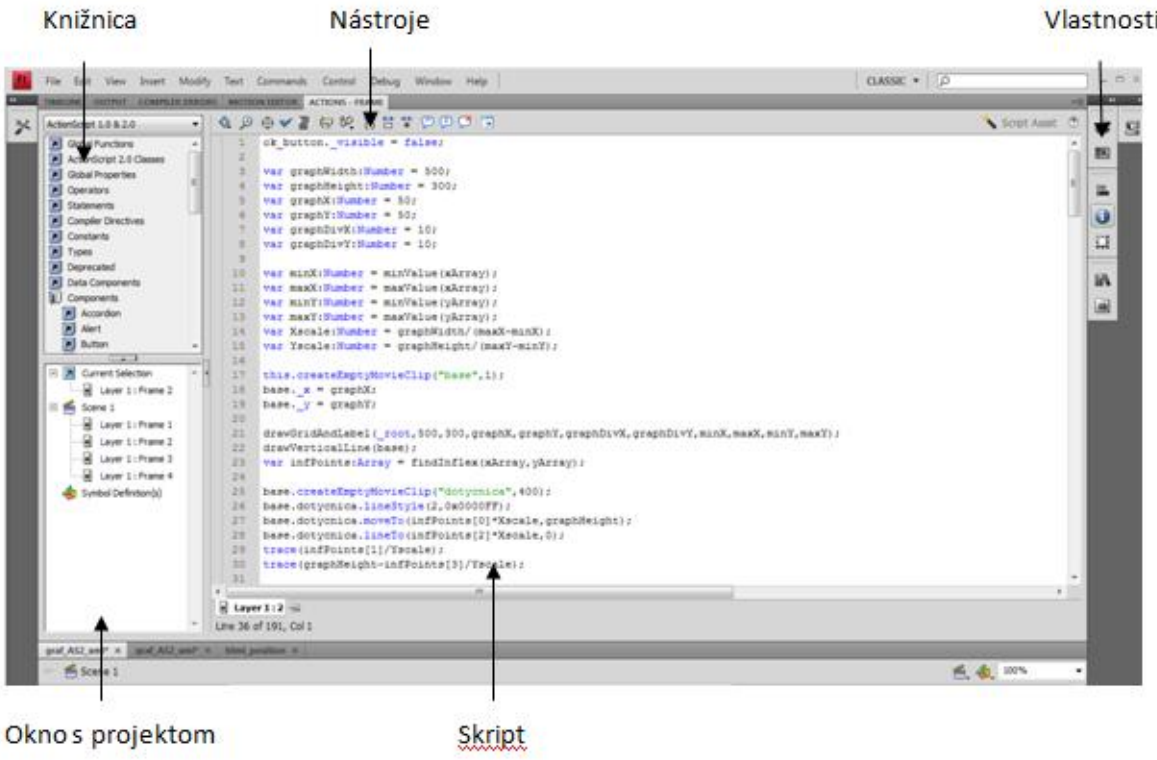
Bitmapová grafika je vlastne súbor bodov, z ktorých sa skladá obrázok. Tieto body majú každý charakterizovaný jas a farbu a splývajú dokopy (Obr. 7). Hlavnou výhodou je všeobecná podpora vo viacerých formátoch. No u bitmapových obrázkoch sa stretávame s problémom dátovej náročnosti. To znamená, že každý bod nesie informáciu o jase, farbe, prípadne iných vlastnostiach o priehľadnosti, tým pádom zaberajú bitmapy veľký úložný priestor na disku.

Obr. 7: Porovnanie bitmapovej a vektorovej grafiky

2.4 ActionScript a jeho verzie

[10]

Flash má tiež vlastný implementovaný jazyk ActionScript (AS). Ide o objektovo orientovaný programovací jazyk využívaný hlavne pri vývoji náročnejších aplikácií a interaktívnych animácií. Slúži na ovládanie jednotlivých objektov, tvorbu formulárov, vytváranie grafov a pod.. Jednou z jeho veľkých výhod je, že netreba vedieť príkazy alebo funkcie naspamäť pretože sa dajú nájsť v prehľadnom zozname, ktorý ponúka vývojové prostredie (Obr. 8). Pomocou AS sa dajú riadiť všetky potrebné dynamické akcie programu.



Obr. 8: Prostredie v AS 2.0

- **knižnica -** obsahuje napr. popis funkcií a syntax jazyka
- **nástroje -** nachádzajú sa tu najčastejšie príkazy požívané pri písaní skriptu
- **vlastnosti** - slúži na zobrazenie základných vlastností dokumentu ako je napr. rozlíšenie, typ použitého AS a pod.
- **okno s projektom -** používa sa pri pohybe medzi jednotlivými snímkami
- **skript -** aktuálny snímok v ActionScripte

AS je aktuálne dostupný v niekoľkých verziách:

ActionScript 1.0 - najjednoduchšia verzia, ktorá sa používa doteraz v niektorých typoch prehrávača Flash Player Lite

ActionScript 2.0 - obsiahlejší oproti verzii 1.0, ideálny pre použitie vo výpočtovo menej náročných aplikáciách napr. vzhľadovo orientovaný obsah ActionScript1.0 a ActionScript2.0 môžu existovať dohromady v jednom FLA súbore

ActionScript 3.0 - je oveľa rýchlejší oproti predchádzajúcim verziám, ale zato si vyžaduje väčšiu znalosť objektovo orientovaného programovania. Plne vyhovuje požiadavkám

ECMAScript umožňuje lepšie spracovanie XML súboru a tiež kvalitnejšiu prácu s obrázkovými elementmi.

```
this.createEmptyMovieClip("krivka", 1);
with (krivka) {
 lineStyle(5, 0xFF0000, 100);
 moveTo(100, 100);
 curveTo(150, 200, 200, 100);
}
```

Obr. 9: Jednoduchý skript na vykreslenie krivky

Obr. 10: Výsledok skriptu

Funkcia *creatEmptyMovieClip* nám vytvorí klip s názvom *krivka*, kde zadefinujeme niekoľko podmienok pre danú krivku ako vidno na obrázku Obr. 9. Tak napr. *lineStyle* nám povie o farbe, hrúbke čiary a priehľadnosti, *moveTo* nás premiestni na nami zadanú pozíciu a posledná funkcia *curveTo* nám zabezpečí ohnutie čiary v požadovaných uhloch (Obr. 10).

2.4.1 Využívané typy dát

[10]

Pre moju prácu som sa rozhodol používať ActionScript 2.0, ktorý rozoznáva nasledujúce dátové typy:

String – predstavuje postupnosť 16 bitových znakov

Number – používaný pre všetky numerické hodnoty

Boolean – jednoduchý dátový typ tvorený hodnotami *true* alebo *false*, ak hodnota nie je inicializovaná tak má hodnotu *false*

Object – ide o dátový typ, z ktorého vychádzajú ostatné komplexné typy

MovieClip – dátovy typ, reprezentujúci vizuálne efekty

TextField – slúži na jednoduché vkladanie textu

XML – objekt pre jazyk XML

Array – lineárne dátové pole

XMLNode – uzol pre XML

LoadVars – používa sa pre spracovanie metód HTTP GET a HTTP POST

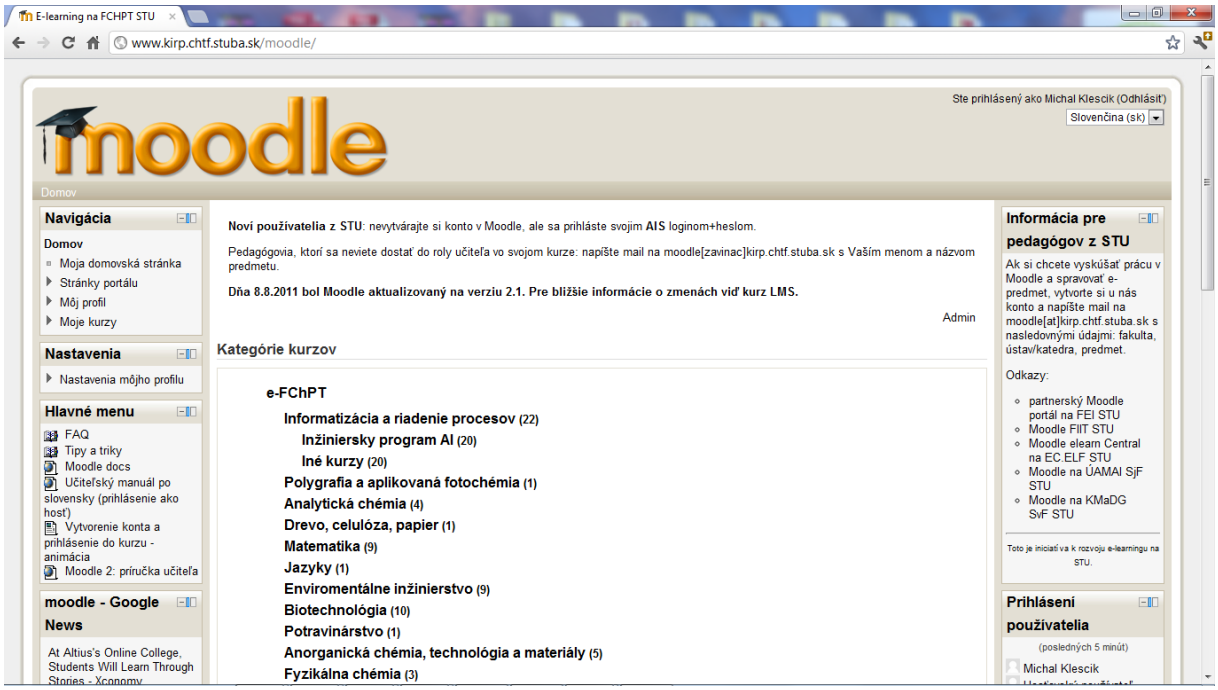
Sound – dátový typ narábajúci so zvukom

NetStream – používa technológiu pre prenos multimediálneho obsahu

Null – využíva sa pre definovanie komplexných dátových typov, obsahuje len hodnotu *null*

2.5 Moodle

Moodle (Modular Object-Oriented Dynamic Learning Environment) je softvérový balíček slúžiaci na vytváranie kurzov dostupných na internete. Prvým jazykom slúžiacim na vytvorenie prototypov bol *Python*, ale neskôr bol prepísaný do jazyka PHP. Prvá verzia bola zverejnená v roku 2002. Slúži hlavne na podporu vyučovania, buď v školskom prostredí alebo aj doma, kde si môže študent nadobudnuté vedomosti upevniť. Do systému sú implementované prezentácie alebo aj samotné video prednášky a testy či už správcom kurzu alebo učiteľom. Za pomoci štandardných modulov je možné priamo vkladať napr. materiály vo forme HTML stránok, flash animácií alebo štruktúrovaných prednášok. Je poskytovaný ako Open Source softvér (pod GNU Public License). Pri dodržaní určitých autorom určených podmienok je možné obsah kopírovať, používať a ďalej šíriť. Tento koncept je určený hlavne pre podporu štúdia vyhovujúceho pre študenta. Moodle môžeme považovať za dobre navrhnutý projekt s pomerne jednoduchou obsluhou a štruktúrou (Obr. 11).



Obr. 11: Moodle systém

3. Praktická časť

V praktickej časti som sa zameral na tvorbu grafov a interaktívnych animácií v programe Flash. Grafy majú slúžiť ako pomocný materiál pri identifikácií systému a získavaní potrebných hodnôt pri výpočtoch. V druhej časti som navrhoval interaktívne animácie, ktoré mali slúžiť k lepšiemu pochopeniu relatívneho a absolútneho poziciovania.

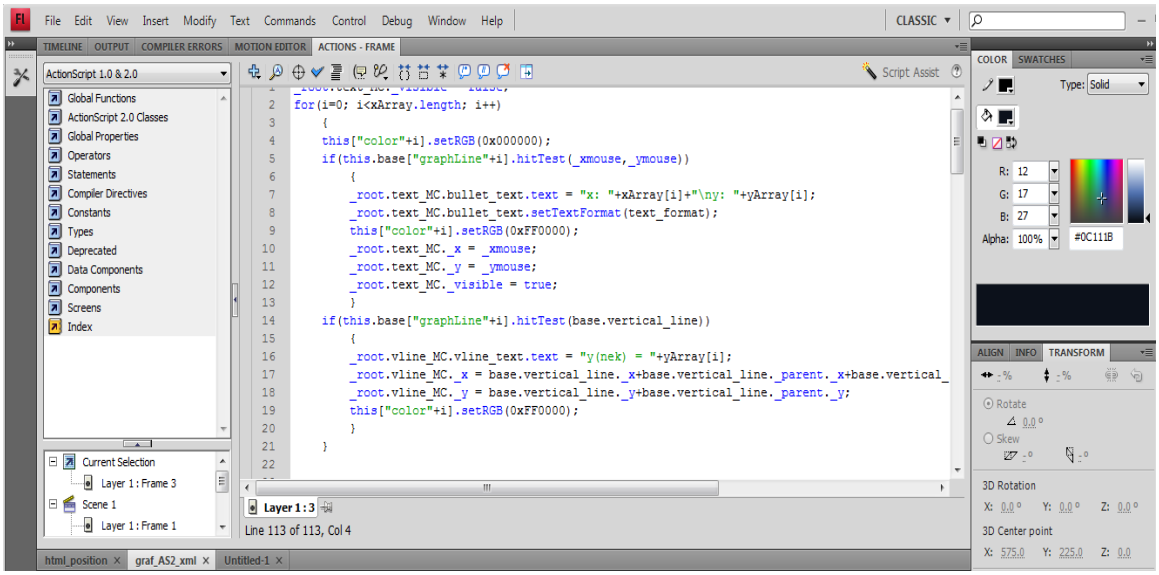
3.1 Programová štruktúra

Pri tvorbe projektu som sa najskôr musel rozhodnúť, ako vlastne chcem aby konečný výsledok vyzeral, a tým pádom si zvoliť podmienky, podľa ktorých budem pokračovať. Celá aplikácia pre tvorbu grafov bola programovo realizovaná spoločne v jednom súbore, preto bolo nutné zvoliť vhodnú štruktúru programu. Program sa tvoril podľa týchto niekoľkých pravidiel :

- snaha o čo najmenšiu veľkosť súboru, ktorú by nám mala zabezpečiť vektorová grafika
- pre programovanie funkcionality sme sa rozhodli využiť ActionScript 2.0
- skript by mal byť štruktúrovaný podľa nejakých všeobecne prijatých pravidiel
- rozlíšenie sme nastavili na hodnoty 600x400 s počtom snímok 30 FPS

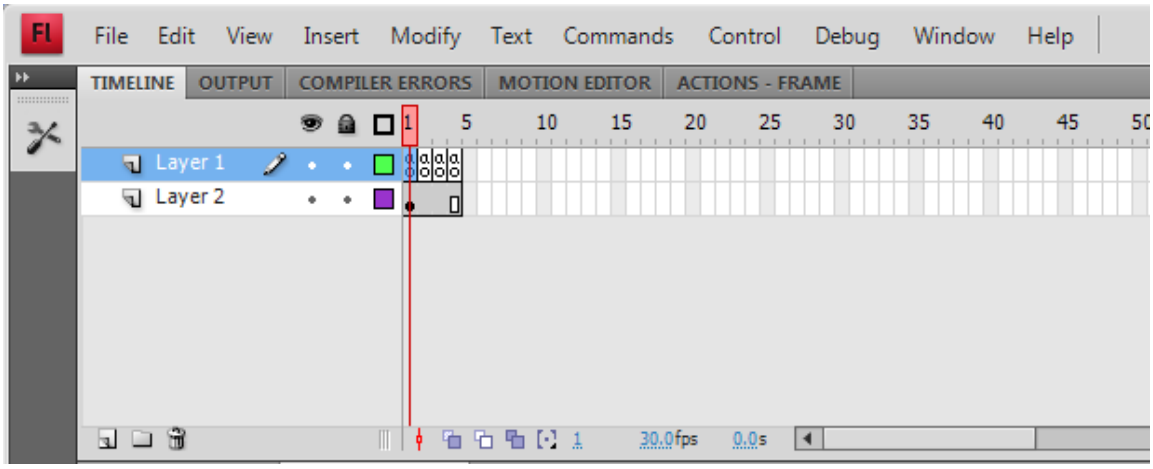
Výsledné Flash projekty som uložil vo FLA súboroch a môžem z nich kompilovať interaktívne SWF aplikácie a exportovať rôzne obrazové a video formáty. Publikovanie v tomto formáte mi zabezpečilo malú veľkosť súboru a je ho možné používať s akýmkoľvek prehliadačom, ktorý má podporu Adobe Flash Player, a ktorý je voľne dostupný na webe. Druhou možnosťou bolo použiť koncovku EXE, kde už samotný súbor poskytuje možnosť na prehratie súborov.

Po rozhodnutí sa ako má vizuálne vyzerať výsledný projekt a s akými koncovkami budem pracovať, som si musel vybrať jednu z možností akú verziu ActionScriptu chcem využiť, keďže ich Flash ponúka hneď niekoľko. Po zvážení náročnosti na pochopenie a zložitosti štruktúry som zvolil *ActionScript 2.0*, ktorý je síce starší, ale vzhľadom na menšiu syntaktickú zložitosť vhodnejší ako *ActionScript 3.0* (*Obr. 12*). Využitie jednotlivých funkcií v zápisoch uvediem nižšie už priamo implementované v skriptoch.



Obr. 12: Prostredie pre AS2.0

Na časovej osi (*Obr. 13*) som si zvolil dve vrstvy. V spodnej vrstve sa nachádza ohraničený priestor, ktorý slúži na vkladanie hodnôt pre vykreslenie príslušných grafov. Hodnoty boli zadávané štandardne podľa osí horizontálnej *x* a vertikálnej *y*. Hodnoty som získaval z grafov pre systémy prvého a vyššieho rádu a následne s pomocou vytvoreného tlačidla *OK* som ich exportoval pre vykreslenie. V prvej vrstve sa nachádzajú samotné *snímky,* kde som pracoval s jednotlivými časťami programu. V prvej *snímke* sa nachádza skript slúžiaci na definovanie premenných *xArray*, *yArray* a *xmlGraph*. *Array* sú vlastne vytvorené polia, kde si definujem hodnoty *x* a *y*, podľa ktorých budú krivky zostrojené. Druhá *snímka* je asi najpodstatnejšia. Nachádza sa v nej všetko spojené s úpravou a vizuálnou stránkou výsledného objektu *Movieclip,* v ktorom bude umiestnený graf. Tretia *snímka* je vlastne doplnkom k druhej a sú tam riešené niektoré detaily vyskytujúce sa v grafoch, napr. zoom určitého bodu alebo vypisovanie *y* hodnoty v blokovom texte.



Obr. 13: Vrstvy na časovej osi

3.2 XML súbory

Po rozhodnutí sa o grafickej podobe projektu, som sa pustil do samotnej tvorby pomocou skriptov. Po zvážení, že namiesto zdĺhavého zadávania hodnôt priamo do rámčeka bolo vhodnejšie využiť dobrú kompatibilitu Flashu s externými súbormi v XML formáte. Je to praktickejšie a navyše je to dobre prehľadný formát (Obr. 14).

```
<?xml version="1.0" encoding="utf-8"?>
<graph>
<data>
        <cas>0</cas>
        <tep>54.90</tep>
</data>
<data>
        <cas>1</cas>
        <tep>54.91</tep>
</data>
<data>
        <cas>2</cas>
        <tep>54.92</tep>
</data>

</graph>
```

Obr. 14: Kompatibilita s XML súborom

Ak som chcel pracovať s XML dokumentom, musel som dodržať niekoľko zásad. Elementy sa musia zakončiť lomítkom a nesmú sa krížiť. Navyše XML má iba jeden hlavný element. Po správnej štruktúre pre XML súbor som do neho zadal jednotlivé hodnoty pre *x* a *y* osi.

Ďalšou úlohou bolo načítanie samotného súboru XML do Flashu. K tomu mi slúžila jednoduchá funkcia load a názov konkrétneho súboru s koncovkou *xml*.

```
xmlGraph.onLoad = function(ok:Boolean)
    {
    if (ok)
      {
      var numOfNodes:Number = this.firstChild.childNodes.length;
      for(var i=0;i<numOfNodes;i++)
        {
        _root.xArray[i]=Number(this.firstChild.childNodes[i].childNodes[0].firstChild.nodeValue);
        _root.yArray[i]=Number(this.firstChild.childNodes[i].childNodes[1].firstChild.nodeValue);
        }

      }
    else
      {
      trace("XML did not load");
      }
    };

xmlGraph.load("graph2.xml");


stop();
```

Obr. 15: Príklad funkcie load

V tomto príklade som si na začiatok zadefinoval premennú *xmlGraph*, ktorej som priradil pomocnú premennú typu *Boolean* a tá mi poskytuje dve možnosti, buď sa požiadavka vykoná čo nám označuje *ok* alebo pri neúspešnom pokuse mi vypíše, že dokument nebol načítaný. V ďalšom kroku som si definoval premennú *numOfNodes* ako číslo, ktoré mi bude slúžiť pre výber konkrétnej dvojice súradníc pre krivku. Keďže som vyberal z pomerne veľa prvkov využil som cyklus *for*, ktorý mi bude vyberať z jednotlivých i prvkov. Vlastnosť *firstChild* mi umožňuje vybrať len jeden prvok z dokumentu a preto som používal ďalšiu z funkcií s názvom *childNodes*. *ChildNodes* je pole obsahujúce zoznam všetkých „potomkov“ nadradeného elementu. Pretože je to pole môžem to využiť pre určenie počtu všetkých child nodes. Nakoniec som načítal pomocou *load* súbor *graph2.xml* kde boli dáta uložené (Obr. 15).

3.3 Vytváranie grafov

Po úspešnom načítaní dát zo súboru som pristúpil k samotnému vytváraniu a vlastnostiam kriviek. Tým pádom, že graf má začiatočný a koncový bod, musel som si vybrať z poľa minimálnu a maximálnu hodnotu pre zabezpečenie správnosti priebehu kriviek (Obr. 16).

```
minValue = function (array:Array)
    {
    var mn:Number = array[0];
    for (i=0; i<array.length; i++)
        {
        if (array[i]<mn && (!isNaN(array[i])))
            {
            mn = array[i];
            }
        }
    return mn;
    };

maxValue = function (array:Array)
    {
    var mxm:Number = array[0];
    for (i=0; i<array.length; i++)
        {
        if (array[i]>mxm && (!isNaN(array[i])))
            {
            mxm = array[i];
            }
        }
    return mxm;
    };
```

Obr. 16: Funkcia pre hranice krivky

Novo vytvorené premenné *mn (minimum)* a *mxm (maximum*) som použil pre podmienku *if*, kde som si zadefinoval aby prvok z poľa bol menší ako *mn* a zároveň ak je to číslo tak nech nám vráti číselnú hodnotu *mn*. To isté som spravil s *mxm* akurát v podmienke som zadal, aby prvok z poľa bol väčší ako *mxm*. Takýmto spôsobom som získal počiatočný a koncový bod. Pokračoval som vytvorením prázdnej inštancie pomocou funkcie *createEmptyMovieClipu,* ktorý som nazval *base* a v ňom sa bude nachádzať krivka s doplnkom pre zobrazenie súradníc. Pre vykreslenie danej krivky som použil skript na *Obr. 17.*

```
for(i=0; i<xArray.length; i++)
    {
    this.base.createEmptyMovieClip("graphLine"+i,i+2);
    this.base["graphLine"+i].lineStyle(2,0x000000, 100, false, "none");

    if(!isNaN(xArray[i]) && !isNaN(yArray[i]) && i>0)
        {
    this.base["graphLine"+i].moveTo(0,0);
    this.base["graphLine"+i].lineTo((xArray[i]-xArray[i-1])*Xscale,-(yArray[i]-yArray[i-1])*Yscale);
    this.base["graphLine"+i]._x = (xArray[i-1]-minX)*Xscale;
    this.base["graphLine"+i]._y = -(yArray[i-1]-maxY)*Yscale;
        }
    this["color"+i] = new Color(this.base["graphLine"+i]);
    this["color"+i].setRGB(0x0000FF);
    }
```

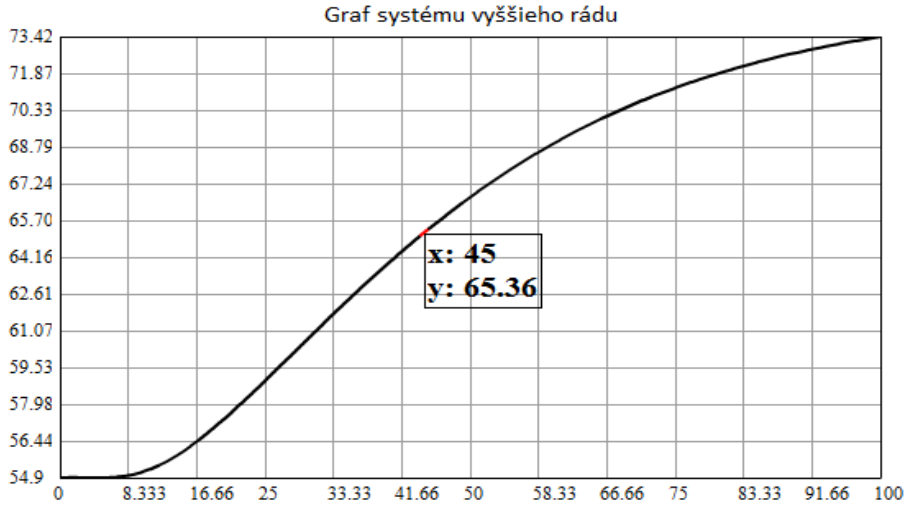
Obr. 17: Funkcie pre vykreslovanie grafov

Na vytvorenie krivky som použil cyklus *for*, v ktorom som si dáta načítal z poľa *xArray*. Vytvoril som si pomocný *movieclip* pomocou *createEmptyMovieClip* s názvom *graphLine*, ktorý som si umiestnil do základného klipu *base*. Vlastnosti krivky mi charakterizuje funkcia *lineStyle*, kde som si zadefinoval niekoľko parametrov napr. hrúbku krivky, farbu krivky, priehľadnosť alebo škálovanie. Príkaz *if* slúži na vyberanie jednotlivých hodnôt z x a y polí. Následne som sa premiestnil pomocou funkcie *moveTo* na počiatočný bod [0,0]. Pomocou funkcie *lineTo* som zapísal vhodný matematický zápis, pomocou ktorého sa krivka krok po kroku vykreslila a následne som musel ošetriť matematicky, aby krivka zostala v požadovanom ohraničenom priestore. Nakoniec som pridal ešte funkciu *hitTest*, ktorá pri premiestnení kurzora na krivku zmenila farbu na červenú. S touto funkciou je spojená ďalšia vlastnosť, ktorú charakterizuje skript na *Obr. 18*.

```
_root.createEmptyMovieClip("vline_MC",5);

_root.vline_MC.createTextField("vline_text", 2, 0, 0, 10, 10);

_root.vline_MC.vline_text.autoSize = true;

_root.vline_MC.vline_text.multiline = true;

_root.vline_MC.vline_text.border = true;

_root.vline_MC.vline_text.selectable = false;
```

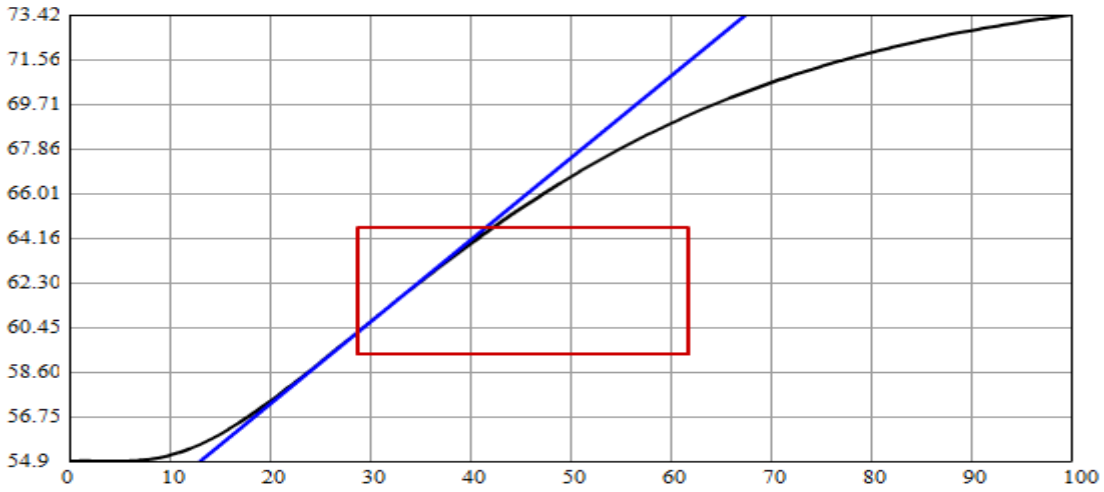
Obr. 18: Charakterizovanie krivky a vlastnosti pre textové pole

Objekt *root* charakterizuje, že sa nachádzam na hlavnej časovej osi resp. v hlavnej časti základného *MovieClipu*. Vytvoril som si ďalší clip s názvom *vline_MC*, do ktorého som vložili textové pole. Tak ako som určil charakteristické vlastnosti pre krivku, tak som zadal aj niekoľko podmienok pre textové pole, napr. hĺbku v akej sa bude nachádzať, výšku a šírku. Definoval som aj niekoľko iných vlastností, pomocou booleanovských hodnôt *true* alebo *false*. Tak napr. jednou z nich je automatická veľkosť alebo orámovanie bloku. Tento textový blok môže byť celkom účinný najmä pri určovaní niektorých metód pri identifikácií, kde nám treba zistiť niektoré zo súradníc na *Obr. 19.*



Obr. 19: Graf systému

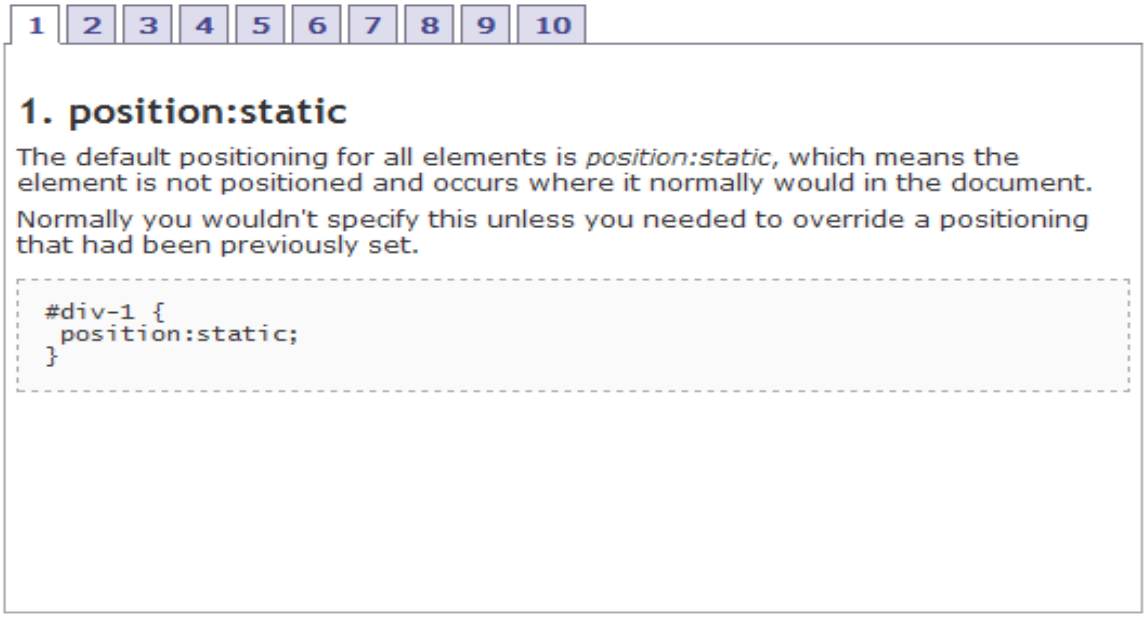
Pretože som pracoval s grafmi, ktoré budú neskôr využité na spracovanie pre identifikáciu rozličných systémov. Bude užitočné pre užívateľa využiť čo najlepšie priblíženie niektorej z oblastí. Je preto vhodné sa zaoberať aj približovaním jednotlivých častí grafu. Pri návrhu ako budem postupovať som sa rozhodol zvoliť si bežnú cestu ako je napr. priblíženie nejakej oblasti na mape na webe. Pomocou kurzora si vyznačím danú oblasť, ktorú chcem priblížiť a po pustení kurzora vidím priblíženú oblasť ako vidieť na *Obr. 20.*



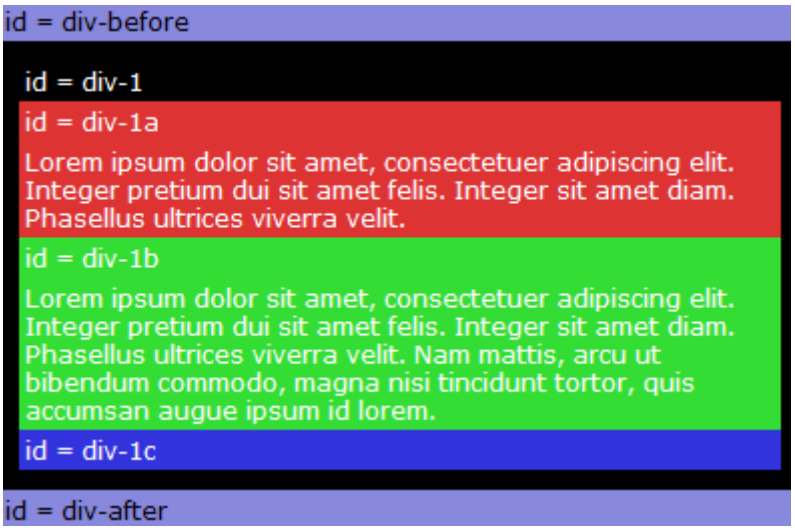
Obr. 20: Označenie oblasti v grafe

3.4 Tvorba animácií

Ďalšou časťou práce bola tvorba interaktívnej prezentácie pre účely výučby predmetu informatizácia a informačné systémy. Obsahom prezentácie sú názorné ukážky využívania kaskádových štýlov CSS pre umiestňovanie statického obsahu HTML dokumentov. V jednotlivých krokoch bolo ukázané ako vhodne použiť umiestňovanie či už statické, absolútne alebo relatívne (*Obr. 21*).



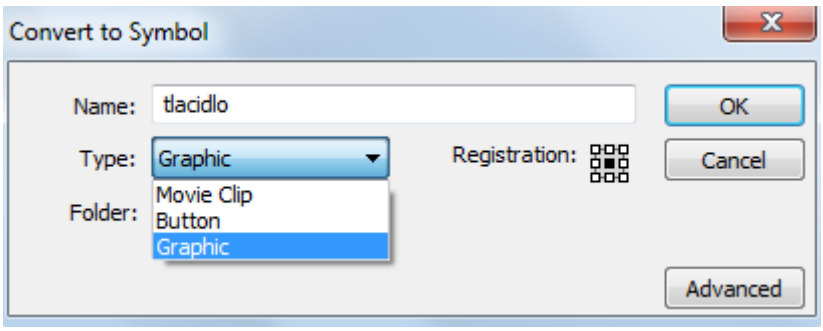
Obr. 21: Pozícia typu static



Obr. 22: Príklad pre statické poziciovanie

Ako je možné vidieť na konkrétnom príklade, na začiatok som si musel zvoliť vhodné parametre hlavného prostredia, v ktorom budem pracovať. Na základe predlohy som začal vytvárať jednotlivé atribúty, aby sa zhodovali s predlohou. Tu je vhodné povedať, že v tejto časti išlo skôr o prácu vo Flashi bez využitia ActionScriptu to znamená kreslenie, vytváranie objektov ručne a ich transformácia na tlačidlá alebo grafické objekty. Na hlavnej scéne som si zvolil tri vrstvy, ktoré som si pomenoval *code, tlacidla a objekty*. Vo vrstve *code* som si zadefinoval funkciu *stop()*, slúžiacu na zastavenie neustáleho behu klipov.

Druhá vrstva s názvom *tlacidla* slúži na prepínanie medzi jednotlivými ukážkami príkladov, ktorých bolo deväť. Ich tvorba bola pomerne jednoduchá, na začiatok som si vytvoril objekt s požadovanými rozmermi a následne som ho pomocou možnosti *Convert to Symbol* predefinoval na požadovanú voľbu. Mal som na výber z troch možností convert to Button, Graphic a MovieClip ako vidno na *Obr. 23*.

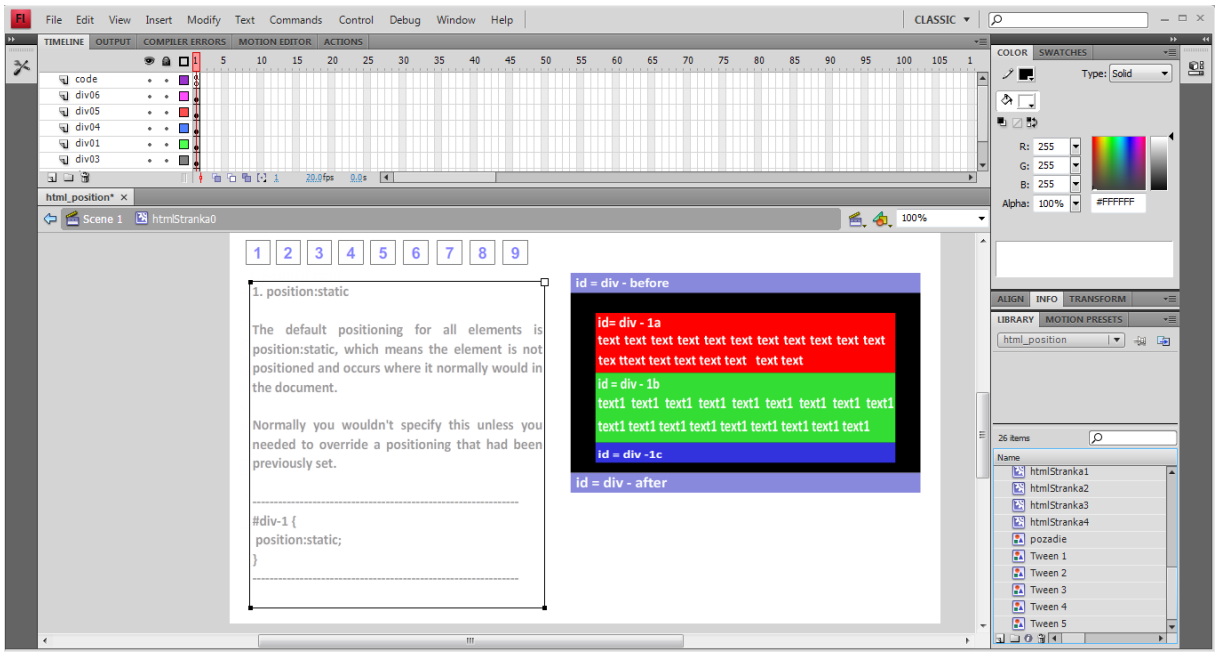


Obr. 23: Konvert objektu na symbol

Pri zvolení možnosti *Button* a následnom transformovaní na *symbol* sa stalo z obyčajného objektu tlačidlo, ktoré má interaktívne vlastnosti. Tvorba ďalších tlačidiel bola realizovaná jednoduchým kopírovaním prvého vytvoreného tlačidla akurát som ich musel premenovať, pretože inak pri snahe zmeniť vlastnosti niektorého z tlačidiel by došlo k automatickej zmene aj ostaných, keďže v knižnici by boli uložené pod rovnakým názvom.

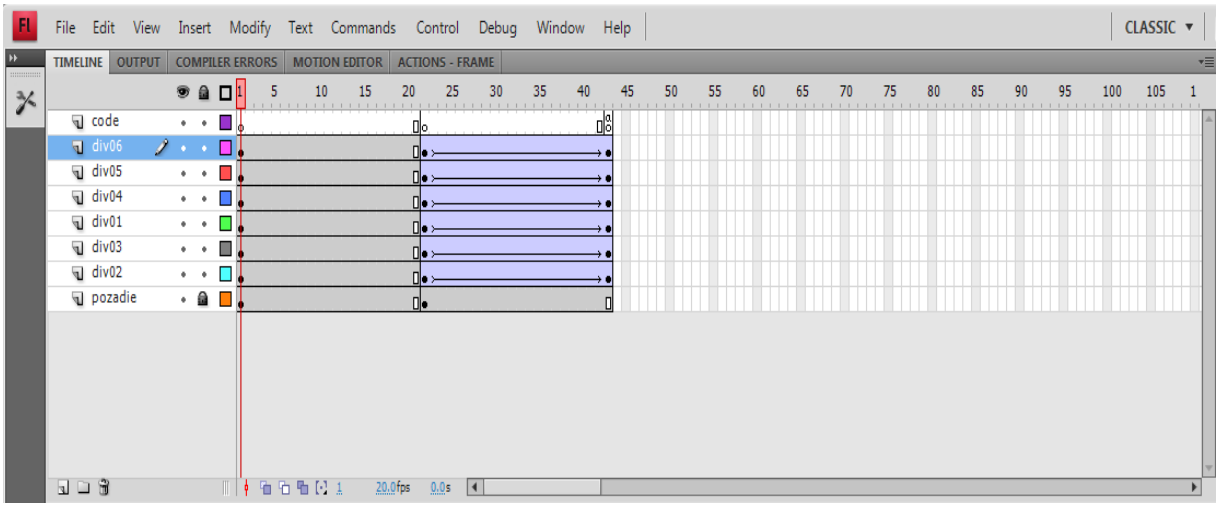
Treťou a zároveň najpodstatnejšou vrstvou je tá s názvom *objekty*. V tejto vrstve som si zvolil deväť kľúčových snímok, v každej som si navrhol obrázok podľa zvolenej predlohy. Jednotlivé objekty som nazval *htmlStranky* číselne od nula až po osem.

Prvá kľúčová snímka predstavuje len statické poziciovanie (*Obr. 24*). To znamená, že prvku nie je možné nastaviť súradnice. Ako prvé som si vytvoril jednotlivé objekty, ktoré som pomenoval samostatne *div*. Každému *divu* som pridelil jeho charakteristické vlastnosti napr. farbu, tvar a umiestnenie v objekte. Keďže ide o spomenuté statické poziciovanie (Obr. 22) objekt sa nebude posúvať žiadnym smerom a ostane na tej pozícii kde som ho vytvoril.



Obr. 24: Kľúčový snímok pre statické poziciovanie

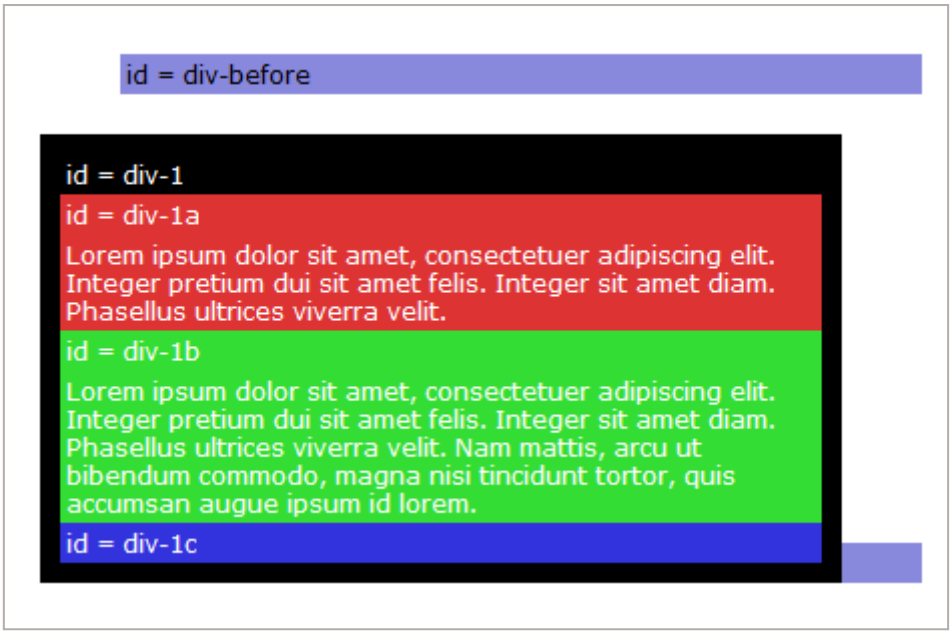
Druhá kľúčová snímka vo vrstve *objekty* predstavuje objekt, ktorému je pridaná relatívna pozícia. To znamená, že prvok sa naformátuje normálne, ale neskôr je umiestnený na zadané súradnice v dokumente. Tak ako v prvom prípade som si vytvoril jednotlivé časti objektu a pomenoval som ich *div* číselne od jedna po šesť (*Obr. 25*).



Obr. 25: Časti objektu pre relatívne poziciovanie

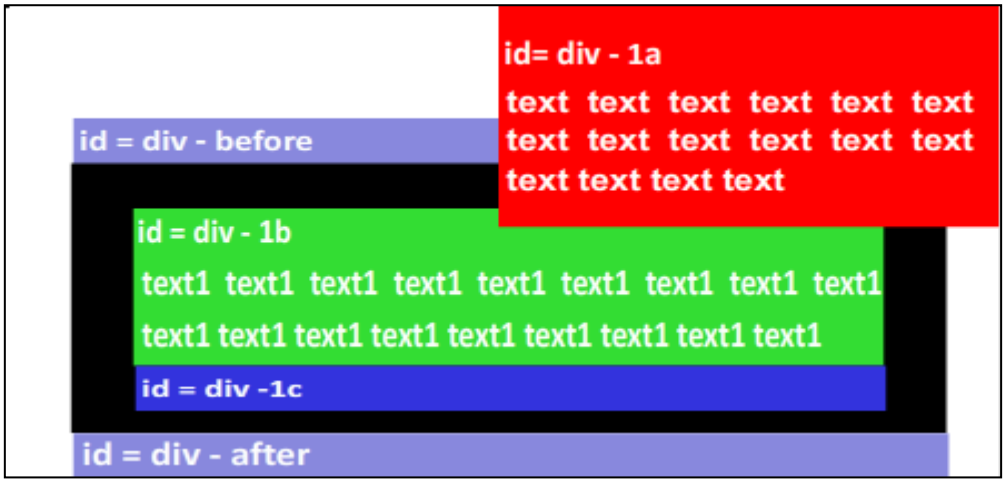
Ako je možné vidieť na časovej osi, od hodnoty jedna až po hodnotu dvadsať som natiahol na začiatok statickú pozíciu. Zvolil som to preto, aby si na začiatok uvedomil pozorovateľ, že objekt je v statickej polohe a až po niekoľkých sekundách dôjde k posunu časti objektu. Posun je realizovaný nasledovne.

Na časovej osi v hodnote dvadsaťjeden som si zvolil nový kľúčový snímok a ďalší som pridal na koniec oblasti, ktorú som si zadal. Realizáciu posunu určitej časti som robil ručne tak, že som zobral daný objekt, ktorý som chcel presunúť a jednoducho som ho premiestnil do zvolenej oblasti. Potom som zvolil na časovej osi vlastnosť *create classic tween*, ktorá zabezpečila pri výslednom sledovaní viditeľnú zmenu z jedného miesta na druhé. Tým pádom som poskytol študentom možnosť interaktívne vnímať relatívne poziciovanie pre nejaký objekt (*Obr. 26*).



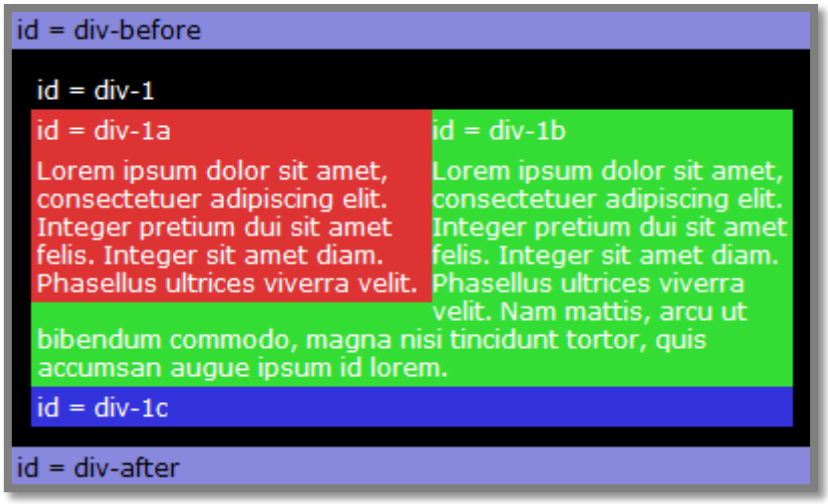
Obr. 26: Relatívne poziciovanie

V tretej kľúčovej snímke som sa zameral na absolútne poziciovanie, čo predstavuje najlepšie riešenie pri určení umiestnenia prvku v rámci stránky. Prvok je zaradený na miesto podľa zadaných súradníc v rámci obmedzujúceho bloku, ktorý predstavuje okno prehliadača. Pri návrhu dynamického posunu som postupoval podobne ako u predchádzajúcej snímky. Po vytvorení jednotlivých častí, som znova použil na začiatok statickú pozíciu pre prvých dvadsať časových úsekov. Následne som presunul červený objekt na zvolené súradnice (*Obr. 27*).



Obr. 27: Absolútne poziciovanie

Avšak pri použití poziciovania môže dizajnér použiť aj kombinácie jednotlivých vlastností napr. relatívne s absolútnym. Treba spomenúť aj ďalšiu vlastnosť, ktorá je pomerne často využívaná a to je takzvané „obtekanie" textu. Obtekanie *float* je vlastnosť, ktorá je vhodná najmä ak sú rozdielne výšky jednotlivý stĺpcov ako vidno na *Obr. 28*. Umožnilo mi to posunúť obrázok alebo text zľava alebo sprava a tak dosiahnem obtekanie textu popri prvom stĺpci.



Obr. 28: Funkcia float

4. Záver

Vhodne vytvorené interaktívne študijné pomôcky s multimediálnymi prvkami dokážu veľmi dobre zefektívniť vyučovací proces. Či už ide o distančnú formu vzdelávania alebo prezenčnú formu. Študijné pomôcky používané pri výučbe poskytujú vyučujúcemu možnosť vytvoriť zaujímavejší spôsob prezentácie a učiteľ má viac času na precvičovanie preberaného učiva a individuálne sa môže venovať jednotlivým študentom.

V prvej časti svojej práce som sa venoval návrhu grafov, ktoré by mali pomôcť pri výučbe predmetu riadenie procesov. Išlo hlavne o čo najjednoduchší spôsob vytvorenia grafu so zadanými hodnotami a príslušnými osami, z ktorých potom študent môže, aj vďaka napr. funkcii "zoom", čo najpresnejšie odčítať potrebné hodnoty pri výpočtoch pre identifikáciu jednotlivých systémov.

V druhej časti som sa zameral na tvorbu dynamických prezentácií ako pomocné materiály pre predmet informácie a informačné systémy. Pracoval som na tvorbe animácií, ktoré mali pomôcť pre lepšie pochopenie práce s kaskádovými štýlmi CSS pre či už statické alebo dynamické umiestňovanie obsahu HTML dokumentov.

Cieľom bakalárskej práce bolo vytvoriť interaktívne pomôcky pre vyučovanie v predmete informácie a informačné systémy. V tejto práci je popísaný princíp tvorby interaktívneho multimediálneho študijného materiálu v programe Flash. Platforma Flash je veľmi vhodným nástrojom pre vytvorenie týchto pomôcok, pretože v ňom vytvorené študijné materiály môžu byť ovládané interaktívne. Materiál môže obsahovať video i zvuk, ale i špičkové učebné animácie, ktoré štúdium môžu urobiť efektívnejším.

5. Zoznam použitej literatúry

[1] Pelůcha, Jiří, Míka, Jiří. Distanční studium v otázkách. Praha: CSVŠ, 2000. 39 s. ISBN 80-86302-16-4

[2] Wikimedia Foundation inc. Free encycklopedia - HTML [online], [cit. 2013-5-20]. Dostupné na internete: <http://en.wikipedia.org/wiki/HTML >

[3] Free Website. W3schools[online] Dostupné na internete: < http://www.w3schools.com/html/html5_intro.asp >

[4] Wikimedia |Foundation inc. Free encycklopedia - JavaScript [online], [cit. 2013-5-20]. Dostupné na internete: < http://sk.wikipedia.org/wiki/JavaScript >

[5] Adobe site Flash CS4 Professional ActionScript 2.0 [online], [cit. 2013-5-20] Dostupné na internete: <http://help.adobe.com/cs_CZ/AS2LCR/Flash_10.0/help.html?content=Part1_Learning_AS2_1.html >]

[6] Milward Brown. Macromedia Flash and Shockwave Players Methodology [online] Dostupné na internete: < http://www.adobe.com/products/player_census/npd/ >

[7] Tomáš Mitana. Flash výhody vs. nevýhody – Flash [online] Dostupné na internete: < http://grafika.sk/clanok/flash-vyhody-vs-nevyhody/ >

[8] Wikimedia Foundation inc. Free encycklopedia –Vektorová grafika [online], [cit. 2013-5-20]. Dostupné na internete: <http://cs.wikipedia.org/wiki/Vektorov%C3%A1_grafika>

[9] Gymnázium Párovská 1 Nitra. Počítačová grafika [online], [cit. 2013-5-20]. Dostupné na internete: <http://www.gymparnr.edu.sk/obsah/predmety/subory/informatika/poc_grafika.pdf>

[10] Wikimedia Foundation inc. Free encycklopedia – ActionScript [online], [cit. 2013-5-20]. Dostupné na internete.: < http://en.wikipedia.org/wiki/ActionScript